APSC 143 – Introduction to Programming for Engineers

Assignment 7 – Searching & Sorting, Memory Management

Figure 1: Mining Trucks. Magnus von Koeller (2007)

Objective:

In this assignment, you will use programming to evaluate multiple payloads from a gold mine. You will determine the mass of the gold ore in each total mass, and its corresponding monetary value. You will then sort this data to have the payload with the highest value at the top and print the sorted data to a file.

Background:

When an ore deposit is mined, only a certain amount of the rock extracted contains the desired mineral. This is called the **ore grade** and can be found using the equation below. The total value of that deposit can be found using the mass of the ore and the unit price of the mineral.

$$G_{ore} = \frac{m_{ore}}{m_{total}}$$

$$v_{ore} = m_{ore} p_{ore}$$

For this assignment, you are given a file called "payloads.txt" which contains a single number on line 1, which represents the number of payloads in the file. The remainder of the file contains the total mass and ore grade of each payload, separated by a new line for each payload.

Instructions:

Below is a guideline for how to approach this problem, though you may choose to approach it however you wish. **Make sure to express your results using the format specified below**:

- Create a file pointer an open the file: **"payloads.txt".**
    - If file opening fails, print "File not found." and end the program.
- Read the first line of the file to determine how many rows of data are in the file.
    - If there are 0 rows of data, print "No payloads in file." and end the program.

- Create a 2D array **using dynamic memory allocation (malloc)** with 3 columns using this data. Read the data from the file into the first two columns of the array.
    - Assume there will always be enough memory to allocate the 2D array.
- Use the equations above to calculate the Ore Value of each payload, and place that data in column 3
    - The Unit price of the Ore is $8500/kg. Masses given are in kg.
- Print the completed array using the format specified below. Make sure your spacing matches the example output given.
- **Sort the array by Ore Value** using any of the methods described in this course.
- Create a new file called **"ores.txt"** and print the sorted array to this file in the same format as before.

Comments are mandatory for this assignment. Add comments as necessary for key pieces in your code, such as variable declaration, conditional statements, and looping conditions to explain what the program is doing.

Your output must match the sample output below exactly; otherwise, the auto grading software will not be able to grade your assignment, which may affect your mark.

Example Outputs:

(Note: Make sure that your spacing matches the example output, and the last line also prints a new line for Mimir to give full credit.)

The correct spacing for the tables are (_**3**_X.XX__**6**___X.XX_**4**__XXXXXXX). Though Gradescope should be lenient

Input File "payloads.txt":

```
7

5434 0.06

6182 0.05

5877 0.04

4895 0.05

5665 0.06

5164 0.03

5376 0.04
```

Console Output:

```
Payload Data:

Total Mass / Grade / Ore Value
```

```
   5434      0.06    $2771340

   6182      0.05    $2627350

   5877      0.04    $1998180

   4895      0.05    $2080375

   5665      0.06    $2889150

   5164      0.03    $1316820

   5376      0.04    $1827840
```

Output File "ores.txt:

```
Sorted Payload Data:

Total Mass / Grade / Ore Value

   5434      0.06    $2771340

   6182      0.05    $2627350

   5665      0.06    $2889150

   4895      0.05    $2080375

   5877      0.04    $1998180

   5376      0.04    $1827840

   5164      0.03    $1316820
```

Submission Instructions:

Create your program using CLion and upload it to Gradescope for grading. **Your program file must be named** "**apsc143assign7.c**" in order for your assignment to be graded. Do not include any personal information (student number, name, etc.) in your submission. Also, please include a comment in your code attesting to the originality of your work.

Refer to the assignment rubric on OnQ for a detailed breakdown of the grading criteria. Your submission must adhere to the assignment rules as outlined in the submission policy document for this course, which can also be found on OnQ. There is zero tolerance for plagiarism in this course. This auto grading software will automatically flag potential cases of plagiarism, which will be reviewed by the instructors.

More information on assignment submissions can be Found in Week 2, and information on the specific definition and repercussions of plagiarism can be found in the "Begin Here (About This Course)" module